混雑緩和問題に対するネットワ-ク最適化 (不確実な環境モデルでの動的行動決定システム)

AUTHOR(S):

清水, 俊之; 田中, 環

数理解析研究所講究録
1048巻 1998年 93-110

混雑緩和問題に対するネットワーク最適化

弘前大学大学院理学研究科　清水 俊之(Toshiyuki Shimizu)
弘前大学理工学部　田中 環 (Tamaki Tanaka)

1 はじめに

「混雑」とは、システム内の人や物の流れにおいて不必要な滞りのことである。本研究の対象である店舗においては、客が購入を希望とする商品を円滑に手に入れることができないような状況のことを言う。この混雑という状況を緩和することを目的とする問題を混雑緩和問題と呼ぶ。

このようなシステムの解析や説明のために、待ち行列ネットワーク理論が用いられる。待ち行列ネットワーク理論では、混雑状況を緩和するため、システム (店舗) 内の人の流れを制御して平均滞在時間を最小化する。店に来る客を制御することは、店側にとっては収益の減少につながるとともに、客は好きなように買物をすることが出来なくなってしまうことになる。

そこで本研究では、全く新しい手法として、店舗をネットワークモデルとしてとらえ、過去のデータに基づいた混雑状況の緩和を提案する。これは、商品棚の配置や商品の陳列のしかたを制御する考えで、店にやって来る客について制御することがないので、店舗のようなシステムにとても適している。

待ち行列ネットワーク理論はシステムの動きの解析が主であり、本研究が対象としている店舗内の混雑状況を改善するような最適化をすることは考えられていない。そこで、第3章で述べるように、店舗内の混雑緩和問題を

1. サービス窓口 (商品棚) の商品を入れ換え、
2. サービス窓口の配置を変える

ことによって、窓口での遅延時間の最小化を行い、システム全体として混雑状況の緩和とする。つまり、問題をサーバの最適配置問題と考えることで混雑状況を緩和する。

2 ネットワーク

第3章で述べるように、本研究の対象とする店舗のモデルはネットワーク構造として表現することができる。ネットワークとは、重み付き有向グラフのことであるが、ネットワークに関連した最適化問題において、その特殊な構造を利用することにより、非常に効率的なアルゴリズムを構成することができる。

この章では、ネットワークの数学的な記述法を与える。店舗をネットワークとみなす際に、その表記が有効な表現方法を与えてくれる。さらに、本研究で重要な役割をする経路や接続関数を定義する。

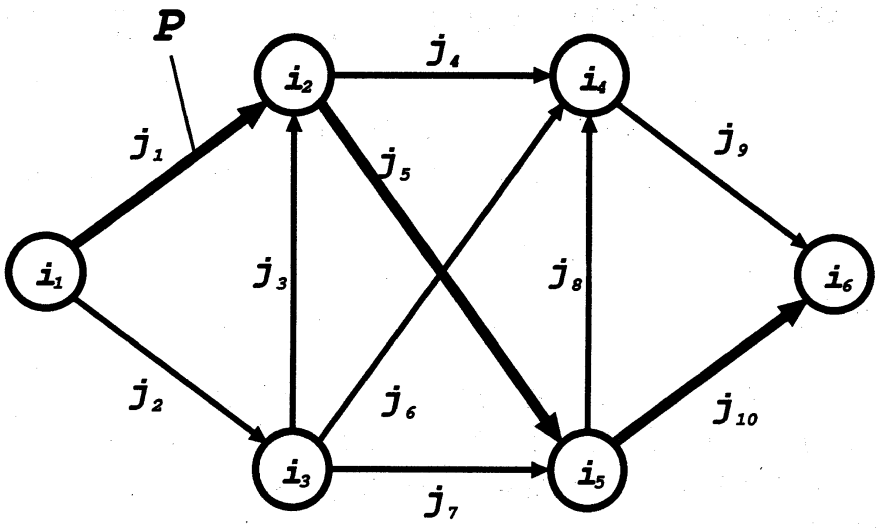

図 2.1: ネットワークの例

2.1 ネットワークの定義

ネットワーク $G = (N, A, \sim)$ を次のように定義する。

定義 2.1 (ネットワーク)
N、A をそれぞれノード (節点) とアーク (枝) の集合とし、次のように定める同値関係$\sim$からなる有向グラフ $G = (N, A, \sim)$ をネットワークと呼ぶ (図 2.1)。一般に、アーク上やノード上にコストなどの属性を与える。
任意のアーク $j \in A$ に対して、同値関係$\sim$を

$$j \sim (i, i') \overset{\text{def}}{\rightleftharpoons} j \text{ はノード } i \text{ からノード } i' \text{をつなぐアークである。}$$

と定義する。ただし、$i, i' \in N, \quad i \neq i'$とする。ノード i, i'をそれぞれアーク j の始点 (initial node)、終点 (terminal node) と呼ぶ。

定義 2.2 (経路)
ネットワーク $G = (N, A, \sim)$ で、ノードとアークを交互に並べた有限列

$$P := i_0, j_1, i_1, j_2, \ldots, j_{r-1}, i_r, \ (r > 0,\ i_{k-1}, i_k \in N,\ j \in A)$$

を経路と言う。ただし、$j_k \sim (i_{k-1}, i_k)$ または、$j_k \sim (i_k, i_{k-1})$ である。

定義 2.3 (接続関数)
ネットワーク $G = (N, A, \sim)$ で、ノード $i \in N$、アーク $j \in A$ に対して、接続関数 $e(i, j)$ を次のように定義する。

$$e(i,j) := \begin{cases} 1, & i \text{ が } j \text{ の始点のとき} \\ -1, & i \text{ が } j \text{ の終点のとき} \\ 0, & \text{その他} \end{cases}$$

2.2 ネットワークの例

例 2.1 ネットワーク

図 2.1 のネットワーク $G=(N,A,\sim)$ で、ノードの集合は $N=\{i_1,\ldots,i_6\}$、アークの集合は $A=\{j_1,\ldots,j_{10}\}$ である。また、アーク j_6 を同値関係$\sim$を用いて表すと $j_6 \sim (i_3,i_4)$ となり、図の経路 P は、

$$P=i_1,j_1,i_2,j_5,i_5,j_{10},i_6$$

と表せる。さらに、接続関数は $e(i_3,j_6)=1,\ e(i_4,j_6)=-1$ のようになる。

道路網や輸送ネットワークのような物理的なモデルだけでなく、状態遷移や論理命題もネットワークとして考えることができる。

3 ネットワークモデル

この章では、2 次元店舗内の混雑緩和問題をモデル化する。まず、本研究において混雑をどのように考えたかを述べ、混雑緩和の手段を提示する。また、次章での定式化に必要な関数などを定義する。さらに、対象とする店舗モデルをネットワーク構造としてとらえる。客の意思決定における店舗内での動きがネットワーク上の経路と同値であることを示し、店舗内の混雑緩和がネットワークの経路上のコスト最小化問題として考えられることを述べる。

3.1 混雑緩和問題

本研究では図 3.1 のような店舗内での混雑緩和を対象とする。一般的な店舗では図 3.1 のように商品棚を配置し、商品を陳列している。しかし、混雑状況を考えたときに、このような配置が本当に良い配置であるかどうか、科学的な根拠や裏付けがあるとは言えない。そこで、商品棚の配置や商品の陳列といったシステムの構造を変えて制御する混雑緩和を提案する。

第 1 章で述べたように、システム内の人や物の流れにおいて不必要な滞りのことを混雑と呼ぶ。本研究が対象とする店舗では、客が購入を希望する商品を円滑に手に入れることができないような状況のことを指すことにする。そこで、混雑状況を緩和する手段を述べる。

定義 3.1 (混雑緩和問題)

システム内の混雑状況を改善する問題を**混雑緩和問題**とする。図 3.1 のような商品棚の配置が最適であるとはいえないので、商品棚の配置を変えることも考慮に入れ、

1. サービス窓口 (商品棚) の商品を入れ換え、
2. サービス窓口の配置を変える

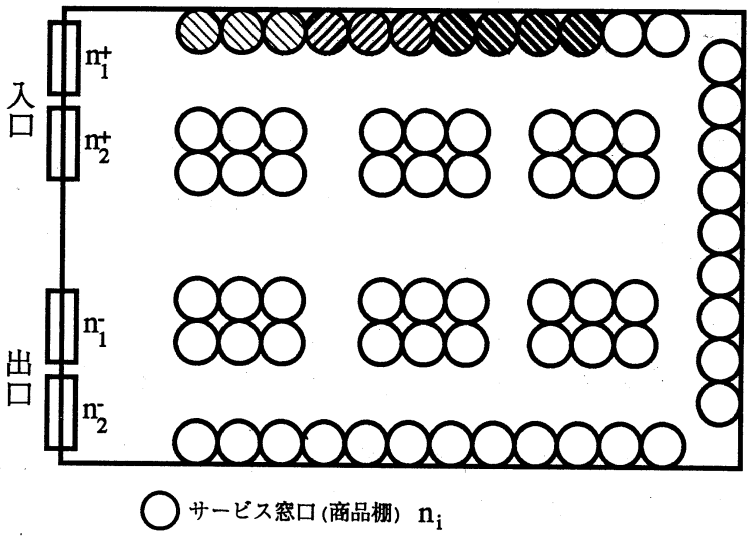

図 3.1: 実際の商品棚の配置の例 (模様は同一商品を示す)

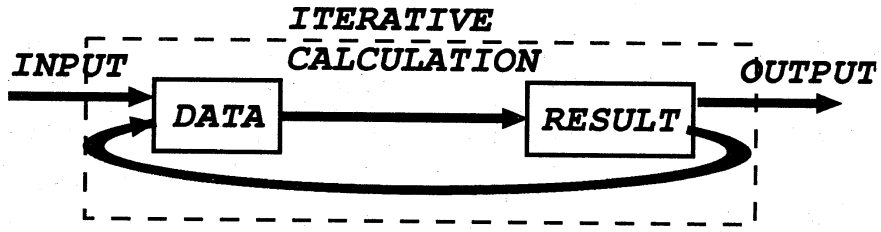

図 3.2: 混雑緩和問題の逐次的な近似計算の基本的な考え方

ことによって、窓口での遅延時間の最小化を行い、システム全体としての遅延時間最小化、つまり混雑の緩和とみなす。

さらに、ある一定期間で得た過去のデータに基づいて計算をし、再び得られた計算結果によって改善するといったような逐次的に解を改良して、近似的に混雑の緩和をする方法をとることにした (図 3.2)。

その理由は、時間の経過により客の動きが変わって行くのに対応するためである。

3.2 モデル化のための想定

混雑緩和問題をモデル化する上で次のようなことを想定する。

- 店舗内には入口 $(n_1^+, \ldots, n_p^+)$ と出口 $(n_1^-, \ldots, n_q^-)$ があり、商品棚 $(n_1, \ldots, n_k)$ に r 種類の商品 $(c_1, \ldots, c_r)$ を陳列することと決める。ただし、陳列されない商品があっては困るので $k \geq r$ とする (図 3.1)。
- $\hat{N}(c_i)$ を商品 c_i の置かれた商品棚の集合とする。
- 客 m_k $(k = 1, \ldots, \ell)$ の購入予定の商品の集合を

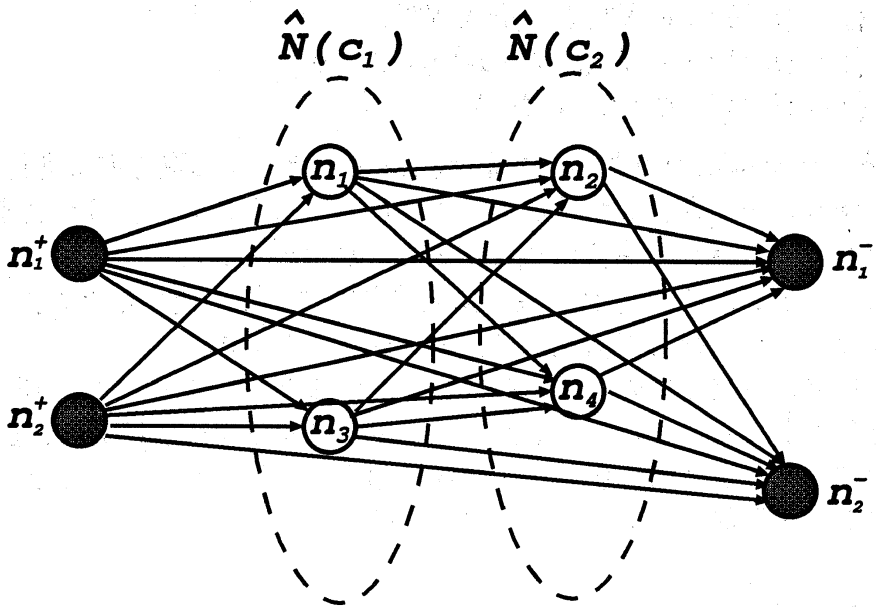

図 3.3: 本研究でのモデル ($p = q = 2, k = 4, r = 2$)

$$D(m_k) := \{c_{\lambda_i} \mid i = 1, \ldots, p,\ \{\lambda_1, \ldots, \lambda_p\} \subset \{1, \ldots, r\}\ \}$$

とする。$D(m_k)$ は時刻によって変化するリストとして考える。なお、客は店舗内に陳列された商品の位置の情報を得ているものとする。

- モデルの簡単化のために、客 m_k の歩く速さは 1 単位時間で座標上の 1 目盛りを移動する速さとする。座標については、定義 3.3 で述べる。また、商品を手に得るのにかかる時間も同様に 1 単位時間とする。

3.3 ネットワークモデル

研究対象の店舗を以下のようなネットワークモデルとして表す。

システム (店舗) 内のサービス窓口 (商品棚)、入口及び出口をノードの集合 N、それらノードを結ぶアークの集合を A とする。さらに同値関係~によってネットワーク $G = (N, A, \sim)$ を定める。

N と A は次のように表すことができる。

$$N := N' \cup N^+ \cup N^-, \quad A := A_1 \cup A_2 \cup A_3$$

ただし、

$$N' := \{n_1, n_2, \ldots, n_k\}, \quad N^+ := \{n_1^+, n_2^+, \ldots, n_p^+\}, \quad N^- := \{n_1^-, n_2^-, \ldots, n_q^-\}$$

とし、それぞれ N' はサービス窓口 (商品棚) の集合、N^+は店舗の入口の集合、N^-は店舗の出口の集合である。

また、

$$A_1 := \{j \sim (i,i') \mid i \in N^+, i' \in (N \setminus N^+)\}$$
$$A_2 := \{j \sim (i,i') \mid i \in \hat{N}(c_\alpha), i' \in \hat{N}(c_\beta), \alpha < \beta\}$$
$$A_3 := \{j \sim (i,i') \mid i \in N', i' \in N^-\}$$

とし、A_1 の各アークは正の向きに一方通行、A_2 の各アークは双方向に通行可能、A_3 の各アークは正の向きに一方通行とする。このように表すことにより、各商品棚と他の商品棚や入口、出口との間の移動を完全に表現することができる。

ここで注意すべきことは、A_2 は商品の陳列の仕方によって変わるので、ネットワーク G は商品の陳列の仕方に依存して構造を変えるということである。

いま定めたモデルは図 3.3 のようになる。一見すると、このモデルが実際の配置とは異なるように感じるかもしれない。しかし、ネットワークはノード間の相対的な関係を表すもので、後で定義するように、各ノードに座標のような属性を与えることにより実際の店舗を表現することが可能となる。

3.4 関数の定義

定式化に必要な関数や行列を定義する。この節で定義するものは、定式化における制約条件や、客の動きを記述する際に重要な役割を果たす。

定義 3.2 (商品の陳列を表す行列)

商品棚 n_i $(i = 1, \ldots, k)$ に陳列されている商品を表す行列を次のように定義する。

$$B := (b_{ij})$$

ただし、$i = 1, \ldots, k$、$j = 1, \ldots, r$ に対して、

$$b_{ij} := \begin{cases} 1, & \text{if } n_i \in \hat{N}(c_j), \quad (\text{商品棚 } n_i \text{ に商品 } c_j \text{ を陳列している。}) \\ 0, & \text{if } n_i \notin \hat{N}(c_j). \quad (\text{商品棚 } n_i \text{ に商品 } c_j \text{ を陳列していない。}) \end{cases}$$

この行列は商品の陳列の変更を定量的に表すことができる。また、行列の各行は 1 が一つで他は 0 であり、各列には必ず 1 が一つ以上である。これが定式化における制約条件の一部になる。

定義 3.3 (商品棚間の距離)

店舗を座標平面上で考える (図 3.4)。各 $i \in N$ に対し、対応する座標 (u_i^1, u_i^2), $u_i^1, u_i^2 \in Z$ を定める。ここで、アーク $j \in A$ 上の距離を表す関数 $d : A \to R$ を次のように定義する (図 3.4 の太線 d)。

$$\forall j \in A \text{ where } j \sim (i,i'), \quad i, i' \in N \ (i \neq i')$$
$$d(j) := \sum_{k=1}^{2} |u_i^k - u_{i'}^k|$$

このように距離を定義した理由は、座標の全てを整数値とすることで、和、差、積が整数値となり、計算上便利だからである。また、格子状の道路設計などに見られるように、このような距離の定義は都市工学においてはよく用いられる。

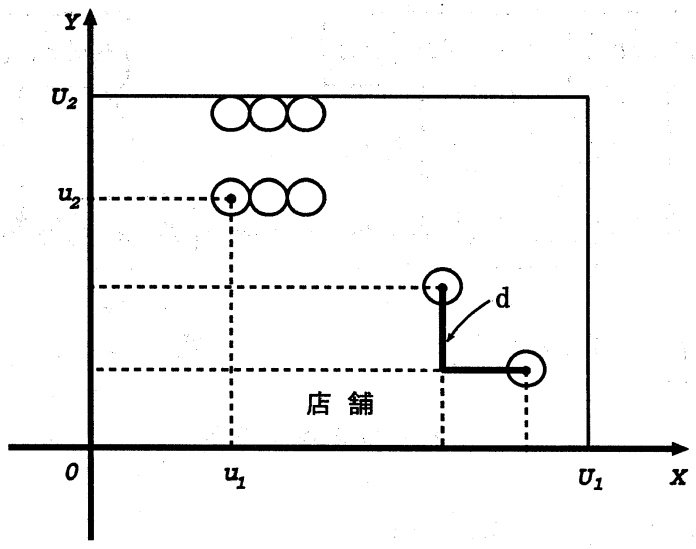

図 3.4: 座標平面上の店舗モデルと距離 d

3.5 客のノード間移動の意思決定

客の動きを定量的に扱うのは、ほぼ困難である。そこで、本研究では次のようなアルゴリズムに従って、客が合理的な判断を下しながらノード間移動の意思決定を行っていると仮定する。この記述によって、各ノード i_ℓへの到着時刻 t_ℓや出発時刻τ_ℓを観測することができる。これらの時刻を各客の全待ち時間を表現する際に用いる。よって、これらの時刻が定式化の目的関数 (システム内の全遅延時間の最小化) の表記において重要である。

アルゴリズム 3.1 (ノード i_ℓ^k から $i_{\ell+1}^k$ への客の動き)

Step 1 : 現在のノード i_ℓ^k から進行可能なアークのリスト A_ℓ^k を以下のように作成する。

(i) $D_\ell^k \neq \emptyset$ のとき、 $A_\ell^k := \{j \in A \mid j \sim (i_\ell^k, i),\ i \in D_\ell^k\}$

(ii) $D_\ell^k = \emptyset$ のとき、 $A_\ell^k := \{j \in A \mid j \sim (i_\ell^k, i),\ i \in N^-\}$

ただし、D_ℓ^k は客 m_k のノード i_ℓ^k での購入予定商品の集合。

Step 2 : i_ℓ^k の次に進むアーク j_l^k を次のように決める。

$\bar{A}_l^k := \{j_{l_p}^k \mid j_{l_p}^k = \arg\min d(j_{l_h}^k)\}$ に対して、

$j_l^k := \arg\min\{N(\tau_l^k) \mid j \in \bar{A}_l^k,\ e(i_l^k, j) = -1\}$

ただし、$N(\tau_\ell^k)$ を時刻τ_ℓ^k でノード i_ℓ^k にいる客の人数とし、混雑度を表す。また、$e(i,j)$ は接続関数である。

Step 3 : 客 m_k がノード $i_{\ell+1}^k$ に到着する時刻 $t_{\ell+1}^k$が次のように決まる。

$$t_{\ell+1}^k := \tau_\ell^k + d(j_\ell^k)$$

Step 4 : ノード $i_{\ell+1}^k$ へ到着した時刻 $t_{\ell+1}^k$ において、混雑度 $N(t_{\ell+1}^k)$ を観測する。$i_{\ell+1}^k$ からの出発時刻が次のように決まる。

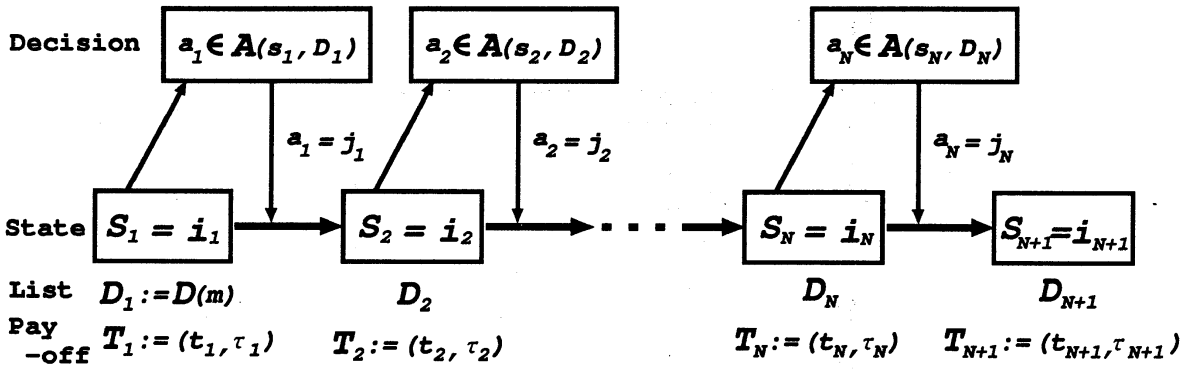

図 3.5: 逐次決定過程

$$\tau^k_{\ell+1} := t^k_{\ell+1} + N(t^k_{\ell+1}) + c$$

ただし、定数 c は商品を得るのにかかる時間で、研究上扱いやすくするために $c := 1$ とする。

アルゴリズムの **Step 1** において、(i) はまだ購入予定商品がある場合で、購入予定商品の入った棚 (ノード) へのアークの集合 A^k_ℓ を生成する。(ii) は購入予定の商品がない場合で、出口までのアークのリスト A^k_ℓ を生成す る。

Step 2 では、**Step 1** で定めた A^k_ℓ のアークの中で、距離の最も短いアークを選ぶ。ただし、そのようなアークが複数存在する場合にはもっとも混雑度の低いノードへのアークを選ぶ。

注意 3.1 (遅延時間)
客 m_k に対して、アルゴリズム 3.1 の **Step 3** や **Step 4** で決まるノード $i^k_\ell \in N$の到着時刻 t^k_ℓや出発時刻τ^k_ℓ によって i^k_ℓ での遅延時間は

$$f_k(i_\ell) := \tau^k_\ell - t^k_\ell - c$$

と表すことができる。

上のように客の動きを定めたことにより、各客がどのように商品棚をまわるかという意思決定は逐次決定過程 (図 3.5) になっている。

各客 m_k $(k = 1, \ldots, \ell)$ に対して、状態 s^k_h $(h = 1, \ldots, N)$ は m_k のいるノード $i^k_h \in N$ を表す。さらに、次にどの商品棚に行くかという行動 a^k_h を $A(s^k_h, D^k_h)$ によって決定し、状態 s^k_{h+1} へ遷移する。また、初期状態 s^k_1 は m_k の到着した店の入口、終端状態 s^k_{N+1} は店の出口を表す。

客 m_k に対して、状態 s^k_h での利得 $T^k_h := (t^k_h, \tau^k_h)$ は、

- t^k_hを決定 $A(s^k_{h-1}, D^k_{h-1})$ によるノード i^k_hへの到着時刻

- τ_h^kを状態 s_h^k、つまりノード i_h^k からの出発時刻

と定める。

客 m_kの意思決定

$$\varphi(m_k) = s_1^k, a_1^k, s_2^k, a_2^k, \ldots, s_N^k, a_N^k, s_{N+1}^k$$

は、状態 $s_1, \ldots, s_{N+1}$と行動 $a_1^k, \ldots, a_N^k$を交互に繰り返す有限列になっている。

注意 3.2 (意思決定とネットワーク上の経路の関係)

各 h, $(h = 1, \ldots, N)$ に対して、客の状態 s_h^k はネットワーク上のノード i_h^k、決定 $A(s_h^k, D_h)$ による行動 a_h^k はアーク j_h^k にそれぞれ対応する。よって、$\varphi(m_k)$ と経路

$$P'_k = i_1^k, j_1^k, i_2^k, j_2^k, \ldots, i_N^k, j_N^k, i_{N+1}^k$$

に対して、

$$\varphi(m_k) = P'_k, \quad \text{for each } k$$

が成立する。

さらに、経路 P'_k に時刻の概念を入れて次のような拡張された経路 P_k を定める。

$$P_k := (t_1^k, i_1^k), (\tau_1^k, j_1^k), \ldots, (t_N^k, i_N^k), (\tau_N^k, j_N^k), (t_{N+1}^k, i_{N+1}^k)$$

そこで、混雑緩和問題は$\varphi(m)$ 上のコスト (遅延時間) の最小化と考えられるが、この性質によってネットワーク上の経路 P におけるコスト最小化問題とみなせる。

4 組合せ最適化問題への定式化

4.1 定式化

これまでに定義した関数等を用いることにより、定義 3.1 の混雑緩和問題は次の最小化問題 **(P)** のようにに定式化することができる。

問題 **(P)** **Minimize**
$$\sum_k \sum_{i \in P_k} f_k(i) \tag{1}$$

Subject to
$$\sum_{j=1}^{r} b_{ij} = 1, \quad \text{for all } i, \tag{2}$$

$$\sum_{i=1}^{k} b_{ij} \geq 1, \quad \text{for all } j, \tag{3}$$

$$U_k^- < u_i^k < U_k^+, \quad \text{for } k = 1, 2, \text{ all } i \in N, \tag{4}$$

$$u_i^1 \neq u_j^1 \text{ or } u_i^2 \neq u_j^2, \quad \text{for all } i, j \in N \ (i \neq j), \tag{5}$$

$$u_i^k, U_k^-, U_k^+ \in Z, \quad \text{for } k = 1, 2, \text{ all } i \in N. \tag{6}$$

(1) 式の目的関数はサービス窓口での遅延時間の最小化を表す。客 m_k のノード (商品棚) $i \in N$ における遅延時間 $f_k(i)$ を前章のように決めたので、これを客 m_k の動き (ネットワーク上での経路 P_k) における全ノードの和をとったものが各客に対する遅延時間と考えることができる。全ての客の待ち時間の和

$$\sum_k \sum_{i \in P_k} f_k(i)$$

を計算することによって、システム全体の遅延時間とみなす。このシステム全体の遅延時間の最小化によって、混雑状況の緩和とみなすこととする。

(2) 式と (3) 式は商品の陳列の仕方に関する制約である。定義 3.2 で定めたように、b_{ij} は商品棚 i に商品 j が陳列されているかどうかを表す関数である。(2) 式は、各商品棚 i に対して、陳列される商品は一種類のみであることを表している。同様に関数の定義から、(3) 式は各商品 j に対して、必ず一つ以上の商品棚に陳列されるように制限している。これは、陳列されない商品が出てくるのを避けるためである。

(4) 式から (6) 式はサービス窓口の配置に対する制約であり、今は長方形の店舗で考える。ただし、店舗の形が複雑であるときは、その形状を示す制約に置き換える必要がある (4.2 節で述べる)。注意すべきは、(6) 式である。各商品棚に対する座標を整数値としている。これは、定義 3.3 で各商品棚間の距離を定義したが、商品棚の座標を全て整数値にすることにより、計算上とても便利で扱いやすいという利点がある。制約条件の (6) 式はこの利点を用いるために加えられる。

4.2 制約条件に対する考察

前節の定式化での制約条件における (4) 式では、対象とする店の形状を長方形で表した。しかし、一般的な店は複雑な形状をしている。その複雑な形状をどのように表現するかをここで考えてみる。例として図 4.1 のような複雑な形状をした店を考える。この例は店舗が凸多面体であるので、制約条件として

$$u_2 \leq F_1(u_1)$$
$$u_2 \leq F_2(u_1)$$
$$u_2 \leq F_3(u_1)$$

のようにすることで店舗の形状を表現することができる。これを問題 **(P)** の制約条件の式 (4) の代わりにすればよい。

また、店舗の形状が曲線になっている場合には、形状を近似する関数で上のように制約条件を構成すればよい。

4.3 組合せ最適化問題とメタヒューリスティックス

有限個の要素からなる実行可能領域の中で目的関数が最小になるような解を見つける問題を組合せ最適化問題という。線形計画問題も有限個の実行可能基底解から最適解を見つ

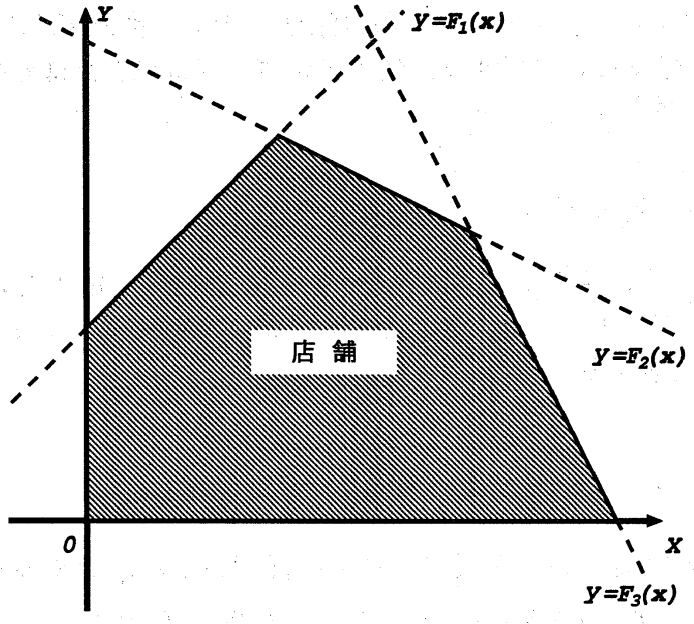

図 4.1: 複雑な形状をした店の例

ける問題であるから、組合せ最適化問題とみなすことができる。最短経路問題、最大流問題や最少費用流問題といったネットワークに関する重要な問題は特殊な線形計画問題なので、やはり組合せ最適化問題である。

4.1 節で定式化した最適化問題 (P) は、組合せ最適化問題になっている。モデル化における仮定において、店舗を座標平面とみなし、商品棚に整数値の座標を与えたことにより有限個の解の中から最適解を見つける問題となっている。

組合せ最適化問題を解くには、有限個の実行可能解を全て調べ、それに対する目的関数値の大小を判断していけば、必ず最適解を見つけることができる。しかし、扱うシステムが大きくなれば実行可能な解の個数は非常に大きくなり、実際に問題を解くのは不可能になってしまう。そこで、効率的な解法が必要となる。

そこで、近年多く用いられている解法にメタヒューリスティックスと呼ばれるいくつかのアルゴリズムがある。タブー・サーチ、シミュレーテッド・アニーリング、さらに遺伝アルゴリズムなどがそれである。

本研究においては、シミュレーテッド・アニーリング法を用いて店舗における混雑緩和問題を解くことを提案する。この方法には、計算時間が長くかかるという欠点がある。しかしながら、次のような店舗を対象とした混雑緩和問題の性質やシミュレーテッド・アニーリング法の優れた性質から、このアルゴリズムがとても都合の良いことが言える。

- 店舗内での混雑を緩和するという本研究の目的において、棚の配置や商品の陳列を変えるという構造の制御は、実際のシステムでは頻繁に行うことができない。なぜなら、頻繁に変えてしまうと、客がどこにどの商品が陳列されているかわからなくなるので、混乱を引き起こすであろう。季節ごとのように、ある長い期間の合間に店舗内の配置・陳列変えを行うということで、計算時間のことは問題にならないであろう。

- シミュレーテッド・アニーリング法は温度パラメータの下げ方をゆっくりとすれば、必ず最適解を得ることができる。他のアルゴリズムはまだ理論的に解明されている部分が多いとは言えない。

4.4 シミュレーテッド・アニーリング法

前節で述べたように、本研究では組合せ最適化問題 **(P)** を解くためのアルゴリズムとして、シミュレーテッド・アニーリング法を用いる。この節では、シミュレーテッド・アニーリング法について説明する。

アルゴリズム 4.1 (シミュレーテッド・アニーリング法)

Step 0 : 凍結温度 $T_{freeze} > 0$ を定め、初期温度 $T > T_{freeze}$ と初期解 x を選ぶ。

Step 1 : 現在の解 x の近傍 $N(x)$ からランダムに解 y を選び、

$$\Delta := f(y) - f(x)$$

とおく。

Step 2 : $\Delta < 0$ ならば x を y で置き換える。

Step 3 : $\Delta \geq 0$ ならば、区間 $[0,1]$ から実数 ξ をランダムに選び、$\xi < e^{-\Delta/T}$ であれば x を y で置き換える。

Step 4 : $T \leq T_{freeze}$ が満たされれば計算終了。そうでなければ、新しい温度 $T_{new} \leq T$ を定める。T を T_{new} で置き換えて、**Step 1** へ戻る。

この方法では、最初の段階では温度を高く設定することにより好ましくない局所最適解に陥ることを避けている。さらに、計算の進行とともに良い解が得られるにしたがって、温度を次第に低下させていくことによって、改悪が起こる確率を下げ、安定した探索を行えるよう工夫している。温度の調整法は計算実験を通して具体的に定めるが、一般的には次のような 2 つのパラメータを定める。

- 温度の減少率を表すパラメータ α $(0 < \alpha < 1, \alpha$ は十分に 1 に近い)
- 各温度に対して何回の反復を行うかを決定するためのパラメータ β $(1 < \beta < 2)$

上のパラメータをもとに、ある温度 T で r 回の反復を行い、それが終われば温度を αT に下げる。さらに新しい温度のもとで行う反復の回数を $\lceil \beta r \rceil$ と変更して、計算を続行する。ただし、実数 a に対して $\lceil a \rceil$ は a 以上の最小の整数を表す。

シミュレーテッド・アニーリング法によって問題を解こうとすればかなりの計算時間を覚悟しなければならないが、それに見合うだけの良い近似最適解を得ることができる。

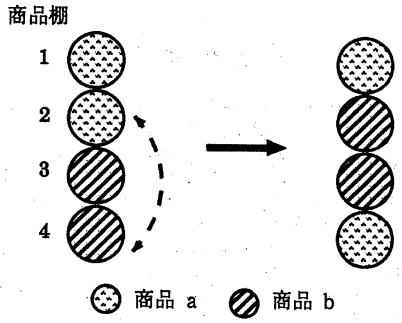

図 4.2: 本研究における近傍 (1) の例

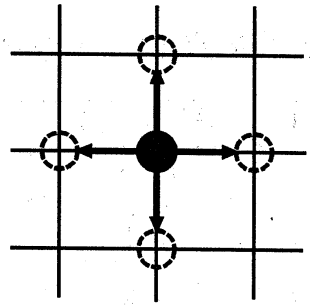

図 4.3: 本研究における近傍 (2)

4.5 本モデルでの適用

シミュレーテッド・アニーリング法を用いて計算するにあたり、本研究で用いる解 x の近傍 $N(x)$ は次のように定義される。

定義 4.1 (解 x の近傍)
本研究の店舗内の混雑緩和問題において、シミュレーテッド・アニーリング法のある反復における解 x の近傍 $N(x)$ を次のように定める。

(1) 商品の 1 回の入れ換え (図 4.2)、

かつ (または)

(2) 商品棚の座標上で 1 の移動 (図 4.3)

を解 x の近傍 $N(x)$ とする。

5 計算例

店舗内の混雑緩和問題に対するモデル化や定式化を行ってきたが、この章では店舗内の混雑緩和の簡単な例を挙げて具体的に考える。

例 5.1
図 5.1 のような店舗での混雑状況の緩和を考える。長方形の店舗に、入口 p_1, p_2、出口 q があり、商品棚 n_1, n_2, n_3 の配置を変えることにより混雑を緩和する。そのためのデータとして、

- 店舗の形状 (図 5.1)
- 商品棚 n_1, n_2, n_3 に陳列されている商品、出入口や商品棚の座標 (緩和前の構造に関するデータ、表 5.1)
- それぞれの客が到着する時刻と入口、購入商品 (客に関するデータ、表 5.2)

表 5.1: 混雑緩和前の店舗のデータ

商品棚	座標 (x, y)	商　品
p_1	$(0, 0)$	—
p_2	$(0, 2)$	—
n_1	$(1, 1)$	c_1
n_2	$(2, 0)$	c_2
n_3	$(2, 2)$	c_2
q	$(3, 1)$	—

表 5.2: 客のデータ

客	到着した 時刻	入口	購 入 商 品
m_1	$t = 0$	p_1	c_1
m_2	0	p_2	c_1, c_2
m_3	1	p_2	c_1
m_4	2	p_1	c_1, c_2
m_5	2	p_2	c_1

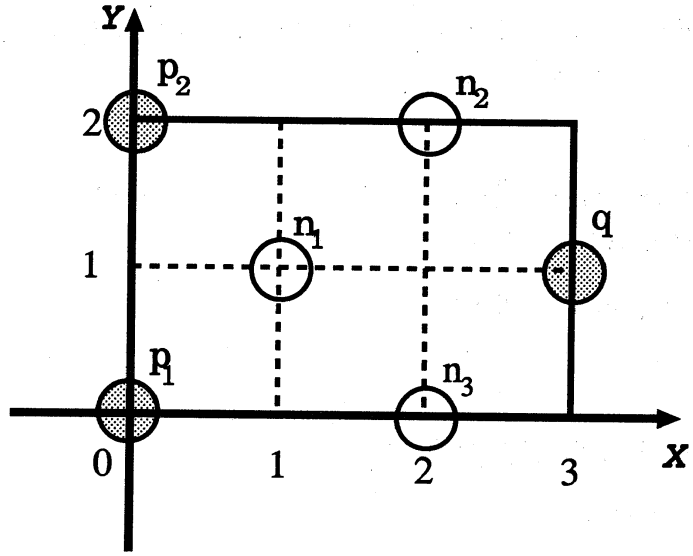

図 5.1: 例 5.1 での店舗 (混雑緩和前)

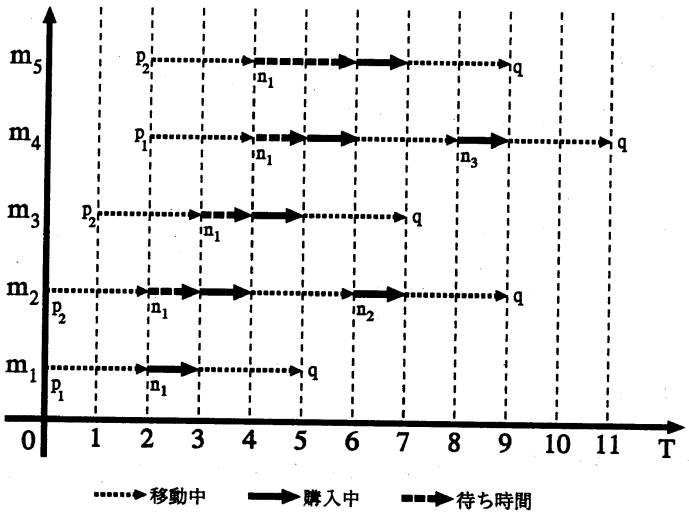

図 5.2: 混雑緩和前の客の動き

を必要とする。

また、定式化において商品の陳列に関する制約条件となる定義 3.2 の商品の陳列を表す行列 $B = [bij] \in R^{3\times 2}$ は、

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

となる。

この例を定式化すると、次のような最適化問題 (P_3) になる。

$$\text{問題 } (P_3) \quad \textbf{Minimize} \quad \sum_{k=1}^{5} \sum_{i \in P_k} f_k(i) \tag{7}$$

$$\textbf{Subject to} \quad \sum_{j=1}^{2} b_{ij} = 1, \quad i = 1,2,3, \tag{8}$$

$$\sum_{i=1}^{3} b_{ij} \geq 1, \quad j = 1,2, \tag{9}$$

$$0 \leq u_i^1 \leq 3,\ 0 \leq u_i^2 \leq 2, \quad \text{for } i = 1,2,3, \tag{10}$$

$$u_i^1 \neq u_j^1 \text{ or } u_i^2 \neq u_j^2, \quad \text{for } i,j = 1,2,3\ (i \neq j), \tag{11}$$

$$u_i^k \in Z, \quad \text{for } k = 1,2, i = 1,2,3 \tag{12}$$

実際に計算すると、混雑緩和前の目的関数値は、

$$\sum_{k=1}^{5} \sum_{i \in P_k} f_k(i) = 5 \tag{13}$$

となり、図 5.2 の太い点線のように合計 5 単位時間の遅延時間が生じている。

この混雑状況を緩和すると、店舗の配置は図 5.3 のようになる。表 5.1 と表 5.3 を比べるとわかるように、商品棚の配置と商品の陳列は図 5.4 のように変わった。各商品棚は座標上の距離で 1 だけ移動しており、さらに商品棚 n_2 の商品が C_1 から C_2 へ変わった。

混雑緩和後の目的関数値は、

$$\sum_{k=1}^{5} \sum_{i \in P_k} f_k(i) = 0 \tag{14}$$

となり、客の流れを制御することなく、店舗の構造を変えることにより混雑状況の緩和をすることができた。

表 5.3: 混雑緩和後の店舗のデータ

商品棚	座標 (x, y)	商　品
p_1	$(0, 0)$	—
p_2	$(0, 2)$	—
n_1	$(1, 0)$	c_1
n_2	$(1, 2)$	c_1
n_3	$(2, 1)$	c_2
q	$(3, 1)$	—

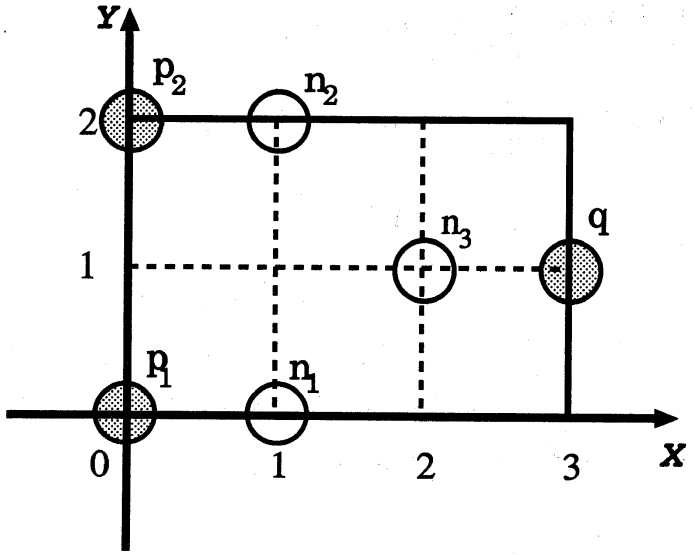

図 5.3: 混雑緩和後の店舗

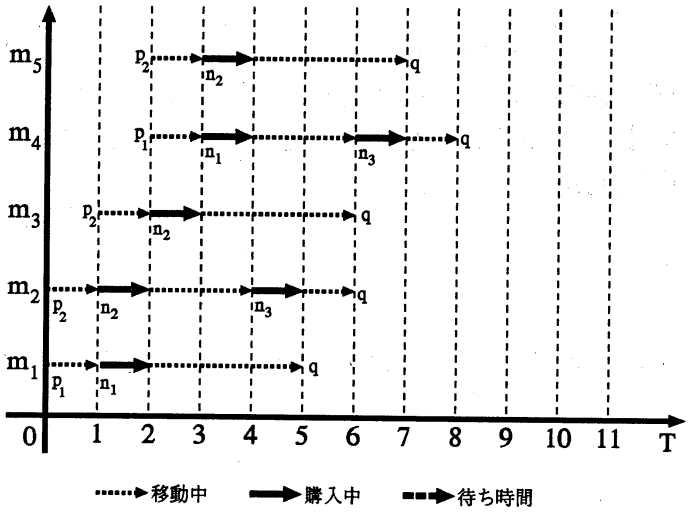

図 5.4: 混雑緩和後の客の動き

表 5.4: 混雑緩和前後の商品棚の配置と商品の陳列

	座標 (x, y)		商品	
商品棚	緩和前	緩和後	緩和前	緩和後
n_1	$(1,1)$	$(1,0)$	c_1	c_1
n_2	$(2,2)$	$(1,2)$	c_2	c_1
n_3	$(2,0)$	$(2,1)$	c_2	c_2

6 まとめ

多くのシステムで生じる「混雑」という状況に着目し、特に店舗を対象とした混雑状況の緩和を行う近似解法を提案した。従来の研究が、混雑状況を緩和するために、システム上の流れを制御していたのに対して、システムの構造を制御することを提案した。

バーゲンセールや特別な時間帯 (昼食時や夕食時など) において、私達は日常的に混雑状況を見ることができる。そこで、客の入場制限といったフローの制御ではなく、店舗内の商品棚の配置や商品の陳列といったシステム構造の制御を考えた。具体的には、過去の客の動きのデータを基にして、

1. サービス窓口 (商品棚) の商品の入れ換え、

2. サービス窓口の配置を変える

ことによって各客の総遅延時間を最小化する最適化問題と考えることができた。

さらに、研究対象である店舗をネットワークモデルとしてとらえることにより、客の店舗内の動きを逐次決定過程に基づくネットワーク上の経路とみなすことができた。したがって、客の意思決定におけるコスト (遅延時間) 最小化問題をネットワークの経路上のコスト最小化問題として定めることができた。ネットワークの離散構造という性質から、問題を組合せ最適化問題として定式化することができた。組合せ最適化問題を解くには、可能な解の組合せを“しらみつぶし的”に調べればいつかは最適解を求めることができる。しかしながら、大規模なシステムにおける解の組合せを全て調べていたのでは、近年の飛躍的な発達したコンピュータの処理能力でさえも莫大な計算時間を必要としてしまう。

そこで、メタヒューリスティックスと呼ばれるアルゴリズムの中から、シミュレーテッド・アニーリング法によって計算することを提案した。この方法で問題を解こうとすれば、かなりの計算時間を覚悟しなければならないが、それに見合うだけの良い解を得ることが期待できる。対象とするモデルが店舗なので、頻繁にシステムの構造を変えることができないという性質から、計算時間のことは問題にならない。

本研究でのモデル化・定式化は、店舗のモデルに限らずいろいろなシステム上の混雑緩和にも適用できるだろう。特に、店舗モデルのように人や物の流れを制御できないようなシステムに対して効果的である。

参考文献

[1] M.O.Ball et al., *Network Models*, Handbooks in Operations Research and Management Science, Vol. 7, North-Holland, Amsterdam, 1995.

[2] Emile Aarts and Jan Korst, *Simulated Annealing and Boltzmann Machines*, John Wiley, New York, 1989.

[3] R.T.Rockafellar, *Network Flows and Monotropic Optimization*, John Wiley & Sons, 1984.

[4] 伊理正夫, 藤重悟, 大山達雄, グラフ・ネットワーク・マトロイド, 講座・数理計画法, 産業図書, 1986.

[5] 伊理正夫, 今野浩, 刀根薫, 最適化ハンドブック, 朝倉書店, 1995.

[6] 伊理正夫, 今野浩, 刀根薫, 確率モデルハンドブック, 朝倉書店, 1995.

[7] 岩本誠一, 動的計画論, 九州大学出版会, 1987.

[8] 清水俊之, 田中環, 弘前市の郵便配達区画改善について, 弘前大学理科報告, Vol.43, pp.313–322, 1996.

[9] T.Shimizu and T.Tanaka, *A Network Optimization for Relief Problem from Congestion*, The Fourth Conference of the Association of Asian-Pacific Operational Research Societies within IFORS (APORS'97) (第4回アジア太平洋地域オペレーションズ・リサーチ学会連合会議), メルボルン, オーストラリア, **(1997,12)**; Session: Network F, Abstracts: p.TC2.3.

[10] 福嶋雅夫, 数理計画入門, システム制御情報ライブラリー15, 朝倉書店, 1996.